18. The method according to claim **1**, wherein said analyzing comprises scanning data files referenced by the object code representing computer program logic to locate data formatted as date data.

19. The method according to claim **1**, wherein said analyzing comprises tracing presumed references to the selected data type through a logical flow of the computer program logic.

20. The method according to claim **1**, wherein the data having the data type is presented to the computer program logic as a set of data records, each data record having a length, said modifying comprising altering a data format without altering a data record length.

21. The method according to claim **1**, wherein said modifying comprises inserting an instruction to temporarily interrupt program logical execution to allow execution of new program logical instructions, following which logical execution of the computer program logic resumes.

22. A method for automatically modifying computer program logic organized as one or more object code modules, comprising the steps of:

(a) analyzing object code logic to identify instances of a desired change, wherein the object code logic is analyzed by one or more processes, selected from the group consisting of:

applying inferential analysis and state dependent analysis to disassembled object code,

scanning data files referenced by the object code to locate data formatted as date data, and

tracing presumed references to the selected data type through a logical flow of the object code;

(b) modifying the object code embodied in the computer program logic with respect to the desired change to modify computer program logical execution, substantially without decompilation or reference to source code; and

(c) storing information representing the modified computer program logic for execution.

23. A method for modifying computer program behavior resolve an ambiguity, comprising:

(a) identifying an ambiguous reference in a computer program;

(b) analyzing the ambiguous reference to determine a likely resolution, wherein the ambiguous reference is analyzed by one or more processes selected from the group consisting of:

applying inferential analysis and state dependent analysis to the computer program, scanning data files referenced by the computer program to locate data formatted as date data, and

tracing presumed references to the selected data type through a logical flow of the computer program;

(c) defining an alteration to the computer program execution to implement the determined likely resolution; and

(d) applying the alteration such that the determined likely resolution is executed during normal execution of the computer program.

24. The method according to claim **23**, wherein said applying comprises applying a patch to said computer program to alter an external reference.

25. The method according to claim **23**, wherein said applying comprises altering a parameter of an operating system under which the computer program executes.

26. The method according to claim **23**, wherein said applying comprises altering externally referenced program logic to selectively execute in a manner dependent on an invoking program.

27. A method for automatically modifying computer program logic, comprising the steps of:

(a) analyzing object code logic of the computer program by applying inferential analysis and state dependent analysis to the object code logic to define instances of a desired change;

(b) modifying the object code in line to effect the desired change in object code logic, substantially without decompilation or reference to source code; and

(c) storing information representing the modified computer program logic for execution.

28. The method according to claim **27**, wherein said analyzing comprises scanning data files referenced by the object code to locate data formatted as date data.

29. The method according to claim **27**, wherein said analyzing comprises tracing presumed references to a selected data type through a logical flow of the object code.

30. A medium for storing a program executable on a computer for automatically modifying object code of another computer program, comprising logical instructions for:

(a) analyzing object code logic of the other computer program by performing an analysis selected from one or more of the group consisting of:

applying inferential analysis and state dependent analysis to the object code logic to define instances of a desired change,

scanning data files referenced by the object code to locate data formatted as date data, and

tracing presumed references to a selected data type through a logical flow of the object code;

(b) modifying the object code to effect the desired change in object code logic, substantially without decompilation or reference to source code; and

(c) storing information representing the modified object code for execution.

31. A computer system for automatically modifying object code of a computer program, comprising:

(a) means for analyzing object code logic of the other computer program by performing an analysis selected from one or more of the group consisting of:

applying inferential analysis and state dependent analysis to the object code logic to define instances of a desired change,

scanning data files referenced by the object code to locate data formatted as date data, and

tracing presumed references to a selected data type through a logical flow of the object code;

(b) means for modifying the object code to effect the desired change in object code logic, substantially without decompilation or reference to source code; and

(c) means for storing information representing the modified object code for execution.

* * * * *